Integrated digital signal processor for parallel asymmetric processing of video and audio multi-media

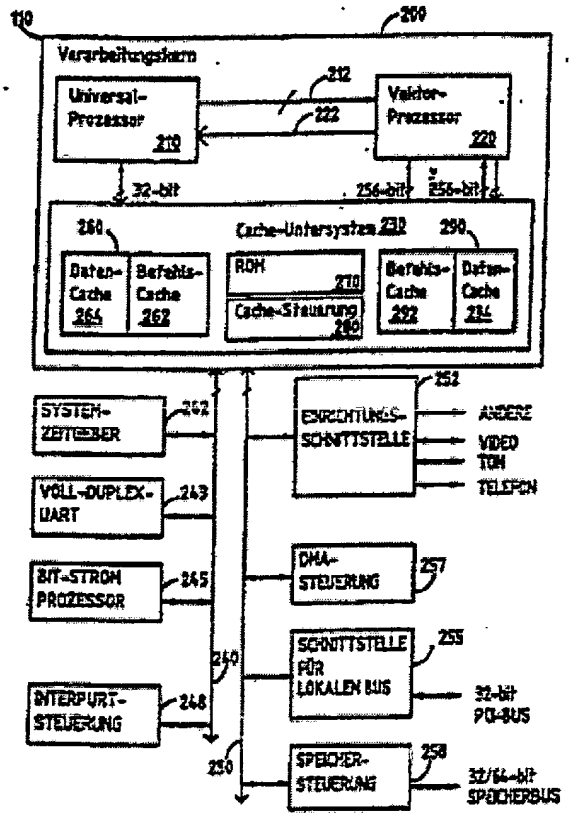
Patent number:	FR2752466
Publication date:	1998-02-20
Inventor:	LE TRONG NGUYEN
Applicant:	SAMSUNG ELECTRONICS CO LTD (KR)
Classification:	
- international:	G06F9/06; G06F15/76
- european:	G06F9/38S4; G06F9/38S6C; G06F15/78V2
Application number:	FR19970010434 19970818
Priority number(s):	US19960697102 19960819

Also published as:

JP10091596 (A)
DE19735981 (A1)

Abstract not available for FR2752466
Abstract of corresponding document: **DE19735981**

The integrated digital signal processor has a general purpose processor (210) and a vector processor (220) for operating in parallel with the general-purpose processor. The general-purpose processor has a set of scalar registers, an instruction decoding unit and an instruction processor (200). The vector processor has a set of vector registers and a second instruction decoder and processor. A cache memory (20) is coupled to the general-purpose processor and to the vector processor. The cache memory has simultaneous read and write ports, and holds a data queue coupled to both processors.



(19) RÉPUBLIQUE FRANÇAISE

INSTITUT NATIONAL
DE LA PROPRIÉTÉ INDUSTRIELLE

PARIS

(11) N° de publication : **2 752 466**
(à n'utiliser que pour les commandes de reproduction)

(21) N° d'enregistrement national : **97 10434**

(51) Int Cl[6] : **G 06 F 9/06**, G 06 F 15/76

(12) DEMANDE DE BREVET D'INVENTION A1

(22) **Date de dépôt** : 18.08.97.

(30) **Priorité** : 19.08.96 US 697102.

(43) **Date de la mise à disposition du public de la demande :** 20.02.98 Bulletin 98/08.

(56) **Liste des documents cités dans le rapport de recherche préliminaire :** *Ce dernier n'a pas été établi à la date de publication de la demande.*

(60) **Références à d'autres documents nationaux apparentés :**

(71) **Demandeur(s) :** *SAMSUNG ELECTRONICS CO LTD* — KR.

(72) **Inventeur(s) :** LE TRONG NGUYEN.

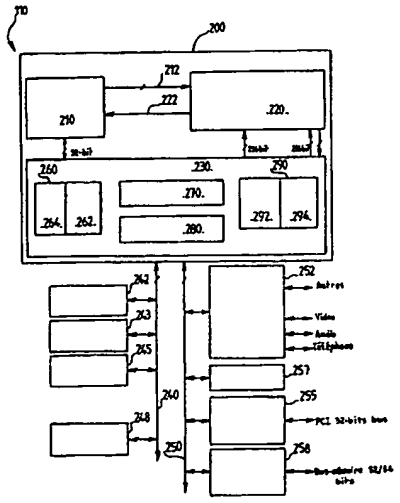
(73) **Titulaire(s) :** .

(74) **Mandataire :** CABINET WEINSTEIN.

(54) DISPOSITIF PROCESSEUR INTEGRE DE SIGNAUX NUMERIQUES.

(57) La présente invention concerne un dispositif processeur intégré de signaux numériques.

Le dispositif est caractérisé en ce qu'il comprend un processeur à tout usage (210) et un processeur de vecteurs (220) pouvant fonctionner en parallèle avec le processeur à tout usage (210). Ce dispositif trouve application dans le domaine du multimédia.



FR 2 752 466 - A1

La présente invention concerne des processeurs de signaux numériques et particulièrement des systèmes de traiement parallèle asymétrique à double mise en place ou à double chemin, qui comprennent un processeur à but général ou à tout usage et un processeur de vecteurs pour manipulation de données vectorielles.

La présente description est en rapport et incorpore à titre de références, dans leur totalité, les demandes de brevets suivantes actuellement déposées :

Demande de brevet US numéro de série non connu, dossier du mandataire US No. M-4355, titrée "Single-Instruction-Multiple-Date Processing in a Multimedia Signal Processor ;"

Demande de brevet US numéro de série non connu, dossier du mandataire US No. M-436555, titrée "Efficient Context Saving and Restoring in Multiprocessors" ;

Demande de brevet US numéro de série non connu, dossier du mandataire US No. M-4366, titrée "System and Method for Handling Software Interrupts with Argument Passing" ;

Demande de brevet US numéro de série non connu, dossier du mandataire US No. M-4367, titrée "System and Method for Handling Interrupts and Exception Events in an Asymmetric Multiprocessor Architecture" ;

Demande de brevet US numéro de série non connu, dossier du mandataire US No. M-4368, titrée "Methods and Apparatus for Processing Video Data" ;

Demande de brevet US numéro de série non connu, dossier du mandataire US No. M-4369, titrée "Single-Instruction-Multiple-Data Processing Using Multiple Banks of Vector Registers" ; and

Demande de brevet US numéro de série non connu, dossier du mandataire US No. M-4370, titrée "Single-Instruction-Multiple-Data Processing with Combined Scalar/Vector Operations".

Une variété de processeurs de signaux numériques sont utilisés dans des applications multimédia telles que

le codage et le décodage de données vidéo, audio et de communications. Un type de processeur de signaux numériques a du matériel dédié pour s'adresser à un problème spécifique tel que décodage ou encodage vidéo MPEG. Des processeurs de signaux numériques à matériel dédié fournissent généralement une performance élevée par coût mais sont seulement utilisables pour des problèmes spécifiques et incapables de s'adapter à d'autres problèmes ou changements en standards.

Les processeurs de signaux numériques programmables exécutent des programmes qui résolvent des problèmes multimédia et fournissent une plus grande flexibilité que les processeurs de signaux numériques à matériel dédié parce que changer un logiciel pour un processeur de signaux numériques programmable peut changer le problème résolu. Un inconvénient des processeurs de signaux numériques programmable est leur plus faible performance par coût. Un processeur de signaux numériques programmable a typiquement une architecture similaire à celle d'un processeur à but général et une puissance de traitement relativement faible. La puissance de traitement faible résulte généralement d'une tentative pour minimiser les coûts. Ainsi, un tel processeur de signaux numériques n'est pas complétement satisfaisant car un processeur de signaux numériques de faible puissance retarde la possibilité du processeur de signaux numériques à s'adresser aux problèmes multimédia plus complexes tels que l'encodage et le décodage vidéo en temps réel.

Puisqu'un but d'un processeur de signaux numériques programmable est de fournir une puissance de traitement élevée pour s'adresser à des problèmes multimédia à un coût minimum, on pourrait l'incorporer dans un traitement parallèle à processeur de signaux numériques, qui est une manière connue d'augmenter la puissance de traitement. Une architecture pour traitement parallèle est un processeur de signaux numériques à "mot d'instruction

très long" (VLIW), qui est caractérisé par un grand nombre d'unités fonctionnelles, dont la plupart accomplissent des taches différentes mais relativement simples. Une instruction unique pour un processeur de signaux numériques à VLIW peut être de 128 octets ou plus et a des parties séparées. Chaque partie peut être exécutée en parallèle par des unités fonctionnelles séparées. Les processeurs de signaux numériques à VLIW ont une puissance de calcul élevée parce qu'un grand nombre d'unités fonctionnelles peuvent fonctionner en parallèle. Les processeurs de signaux numériques à VLIW ont également un coût relativement faible parce que chaque unité fonctionnelle est relativement petite et simple. Un problème pour les processeurs de signaux numériques à VLIW, cependant, est l'inefficacité dans la gestion de commande d'entrée/sortie, la communication avec un ordinateur hôte et autres fonctions qui ne conduisent pas elles-mêmes à une exécution parallèle dans les unités fonctionnelles du processeur à signaux numériques à VLIW. Additionnellement, des programmes pour VLIW diffèrent des programmes d'ordinateur conventionnels et peuvent être difficiles à exécuter à cause du manque d'outils de programmation et de programmeurs familiers aux architecteurs à logiciel à VLIW.

Selon l'invention, un processeur de signaux numériques intégré est décrit. Le processeur de signaux numériques combine un processeur à tout usage avec un processeur de vecteurs, qui est capable de fonctionner en parallèle avec le processeur à tout usage. Le processeur de signaux numériques intégré est capable d'accomplir une performance élevée à faible coût puisque les deux processeurs accomplissent seulement des taches idéalement adaptées pour chaque processeur. Par exemple, le processeur à tout usage fait tourner un système de fonctionnement en temps réel et accomplit une gestion de système global tandis que le processeur de vecteurs est utilisé pour accomplir des calculs parallèles en

utilisant des structures données appelées "vecteurs". Un vecteur est une collecte d'éléments de données typiquement du même type.

Dans un mode de réalisation, le processeur de signaux numériques comprend également un sous-système antémémoire ou à mémoire cache, un premier bus et un second bus. Le premier bus est utilisé pour des dispositifs à vitesse élevée tels qu'une interface de bus local, un contrôleur d'accès direct en mémoire (DMA), un contrôleur de dispositif et un contrôleur de mémoire. Le second bus est utilisé pour des dispositifs à vitesse lente tels qu'une temporisation de système, un UART, un processeur de train de bits et un contrôleur d'interruption.

Le sous-système antémémoire combine des fonctions antémémoire à des fonctions de tableau de commutation ou de routage de données. Les fonctions de tableau de commutation ou de commutateur permettent des trajets de communication multiples entre les processeurs et les bus pour fonctionner simultanément. De plus, la portion antémémoire du sous-système antémémoire permet des lectures et des écritures simultanées dans la mémoire cache.

L'invention sera mieux comprise et d'autres buts, caractéristiques détails et avantages de celle-ci apparaîtront plus clairement dans la description explicative qui va suivre faite en référence aux dessins schématiques annexés donnés uniquement à titre d'exemple illustrant plusieurs modes de réalisation de l'invention et dans lesquels :

- la figure 1 représente un schéma blocs d'une carte multimédia selon un mode de réalisation de l'invention ;

- la figure 2 représente un schéma blocs d'un processeur de signaux multimédia selon un mode de réalisation de l'invention ;

- la figure 3 illustre des relations entre des processeurs et un logiciel ou micrologiciel dans un système comprenant un processeur multimédia selon un mode de réalisation de l'invention ;

- la figure 4 représente un schéma blocs d'un sous-système antémémoire selon un mode de réalisation de l'invention ;

- la figure 5 représente une topographie de mémoire selon un mode de réalisation de l'invention ;

- la figure 6 représente un schéma blocs d'une file d'attente de données utilisées dans un sous-système antémémoire selon un mode de réalisation de l'invention ;

- la figure 7 représente un schéma blocs d'une seconde file d'attente de données utilisées dans un sous sytème antémémoire selon un mode de réalisation de l'invention ; et

- la figure 8 représente un schéma blocs d'une file d'attente d'adresse utilisée dans un sous-système antémémoire selon un mode de réalisation de l'invention.

L'utilisation des mêmes symboles de référence aux différentes figures indique des éléments similaires ou identiques.

Selon un aspect de l'invention, un processeur multimédia comprend un processeur à tout usage et un processeur de vecteurs qui fonctionnent en parallèle selon des chemins de programme séparés. Le processeur à tout usage, comme la plupart des processeurs à tout usage conventionnels, exécute des instructions qui manipulent typiquement des données scalaires. De tels processeurs sont adaptés pour l'exécution de fonctions d'entrée/sortie (I/O) et de commande de contrôle. Dans certains modes de réalisation, le processeur à tout usage a une capacité de traitement de vecteurs limitée de plusieurs éléments de données de dimensions d'octets groupées dans un mot de données. Par exemple, si le processeur à tout usage est un processeur 32 bits, certains modes de réalisation du processeur à tout usage

peuvent traiter simultanément quatre éléments de données d'un octet. Cependant, des calculs multimédia tels que compression et décompression de données audio et vidéo exigent beaucoup de calculs répétitifs sur des groupes d'éléments d'image et des chaînes de données audio. Pour accomplir des opérations multimédia en temps réel, un processeur à tout usage qui manipule des données scalaires (par exemple une valeur d'élément d'image ou une amplitude de son par opérande) ou seulement de petits vecteurs doit fonctionner à une fréquence d'horloge élevée. Au contraire, le processeur de vecteurs exécute des inscriptions où chaque opérande est un vecteur contenant des éléments de données multiples (par exemple des valeurs d'éléments d'image multiples ou des amplitudes de son). De ce fait, le processeur de vecteurs peut accomplir des opérations multimédia en temps réel à une fraction de la fréquence d'horloge exigée pour un processeur à tout usage d'accomplir la même fonction.

Ainsi, en autorisant une division performante des taches exigées pour une application multimédia, la combinaison d'un processeur à tout usage et d'un processeur de vecteurs programmables fournit une performance élevée par coût. Dans un mode de réalisation de l'invention, le processeur à tout usage exécute un système fonctionnant en temps réel conçu pour une carte imprimée média ("carte") communiquant avec un système d'ordinateur hôte. Le système fonctionnant en temps réel communique avec un processeur primaire du système d'ordinateur, des disposifs I/O de services sur la carte ou couplés à la carte et choisit des tâches que le processeur de vecteurs exécute. Dans ce mode de réalisation, le processeur de vecteurs est conçu pour accomplir des tâches de calculs intensifs exigeant une manipulation de blocs de données importants, tandis que le processeur à tout usage agit comme le processeur maître au processeur de vecteurs. Des chemins de programme pour chaque processeur sont écrits en utilisant

un jeu d'instructions conventionnel qui rend le processeur multimédia "très facile à utiliser pour le programmeur". La programmabilité permet au processeur multimédia d'accomplir une variété de taches multimédia différentes. Le processeur multimédia peut, par exemple, être adapté à un nouveau protocole en changeant simplement soit ses programmes d'application soit son micrologiciel. Dans un mode de réalisation, le jeu d'instructions est similaire à celui d'un jeu d'instructions RISC conventionnel (ordinateur à jeu d'instructions réduit).

Selon un autre aspect de l'invention, le processeur à tout usage et le processeur de vecteurs partagent une variété de ressources sur puce et hors puce qui sont accessibles par un espace d'adresse unique. Un sous-système antémémoire qui met en application des caches de données et d'instructions séparés pour chaque processeur fournit également une connexion du type commutateur entre une mémoire locale et des ressources tels qu'un processeur à flux de bits, une interface de transmission asynchrone ("UART"), un contrôleur d'accès direct en mémoire ("DMA"), une interface de bus local et une interface codeur-décodeur ("CODEC") qui sont des dispositifs cartographiés de mémoire. Le sous-système antémémoire peut utiliser un protocole à transaction orientée qui met en oeuvre un commutateur pour accès de données parmi les processeurs et les ressources cartographiés de mémoire.

La figure 1 représente une carte multimédia 100 selon un mode de réalisation de l'invention. La carte multimédia 100 comprend une plaque à circuit imprimé, un processeur multimédia 110 et un connecteur qui se fixe à un bus local 105 d'un système d'ordinateur hôte. Dans un mode de réalisation à titre d'exemple, le bus local 105 est un bus PCI ; mais dans d'autres modes de réalisation, le bus local 105 pourrait être un bus exclusif ou un bus

qui se conforme à tout protocole souhaité tel que les protocoles de bus ISA ou VESA.

Le processeur multimédia 110 utilise une mémoire locale 120, également située sur la carte multimédia 100, pour mémoriser des instructions de données et des programmes. La mémoire locale 120 peut agir également comme une mémoire d'image pour des applications de codage et de décodage vidéo. Dans le mode de réalisation à titre d'exemple, la mémoire locale 120 peut être mise en oeuvre par une mémoire à accès aléatoire dynamique synchrone de 32 bits à 512K (DRAM). Des portions de l'espace de mémoire locale peuvent également être mises en oeuvre par une mémoire à accès aléatoire statique sur pastille ("SRAM") et une mémoire morte ("ROM") dans le processeur multimédia 110. En fait, si une mémoire sur pastille ou puce est prévue pour maintenir suffisamment les données et les instructions de la carte multimédia 100, la mémoire locale 120 n'a pas besoin d'être implantée.

En plus du processeur multimédia 110 et de la mémoire locale 120, la carte multimédia 100 comprend un convertisseur vidéo analogique-numérique 132 (ADC), un convertisseur vidéo numérique-analogique 134 (DAC), un ADC audio 142, un DAC audio 144, un ADC de communications 146 et un DAC de communications 148. Chacun des convertisseurs 132, 134, 142, 144, 146 et 148 peut être mis en oeuvre par un ou plusieurs circuits intégrés séparés. Alternativement, deux ou plus des convertisseurs 132, 134, 142, 144, 146 et 148 peuvent être intégrés sur un circuit intégré unique. Par exemple, un circuit intégré unique 140, par exemple, le AD1843 disponible chez Analog Devices, Inc., peut mettre en oeuvre les fonctions des convertisseurs 142, 144, 146 et 148.

Le ADC vidéo 132, qui peut être mis en oeuvre par, par exemple, un circuit intégré KS0122 disponible chez Samsung Semiconductor, Inc., se relie à une caméra vidéo ou autre source d'un signal vidéo et numérise le signal vidéo en une série de valeurs d'éléments d'image. La

carte multimédia 100 comprime ou encode les valeurs d'éléments d'image selon un standard d'encodage vidéo tel que MEPG, JPEG, ou H.324 implantés dans le micrologiciel exécuté par le processeur multimédia 110. Les données vidéo encodées peuvent alors être transmises à l'ordinateur hôte par l'intermédiaire du bus local 105, à un dispositif tel qu'une carte Ethernet couplée au bus local 105 ou être de plus encodées pour transmission sur une ligne téléphonique couplée au DAC de communication 148.

Le DAC vidéo 134 convertit une série d'échantillons numériques du processeur multimédia 110 en un signal vidéo analogique pour un moniteur vidéo ou une télévision. Le DAC vidéo 134 peut être mis en oeuvre, par exemple, par un circuit intégré KS0119 disponible chez Samsung Semiconductor, Inc., selon un standard vidéo NTSC ou PAL. Le processeur multimédia 110 peut produire la série d'échantillons numériques pour le DAC vidéo 134 sur la base de données reçues de l'ordinateur hôte, d'un autre dispositif couplé au bus local 105, d'une caméra vidéo couplée au ADC vidéo 132 ou d'une ligne téléphonique couplée au ADC de communication 146.

Un composant optionnel de la carte multimédia 100 est un contrôleur graphique 150 qui partage la mémoire locale 120 avec le processeur multimédia 110 et fournit un signal vidéo à un moniteur vidéo pour le système hôte. Le contrôle graphique 150 peut être mis en oeuvre, par exemple, par un contrôleur graphique super VGA disponible chez divers vendeurs, tels que Cirrus Logic, S3, ou Trident Microsystems. Le processeur multimédia 110 produit et mémorise des topographies d'éléments d'image dans la mémoire locale 120 de laquelle le contrôleur graphique 150 produit un signal vidéo pour le moniteur vidéo.

Le ADC audio 142 et le DAC audio 144 sont pour l'entrée et la sortie de signaux audio analogiques. Selon un aspect de l'invention, la carte multimédia 100 simule

les fonctions d'une carte de son, telle que la populaire "SoundBlaster" et met en oeuvre des fonctions de synthèse de son telles que synthèse de table d'ondes et synthèse FM. Pour des simulations de carte de son, un programme d'application exécuté par l'ordinateur hôte fournit des données représentant un son et le processeur multimédia 110 produit des amplitudes de son selon ces données. Le DAC 144 convertit des amplitudes de son en un signal audio analogique pour un haut-parleur ou un amplificateur. Le processeur multimédia 110 traite de façon similaire des signaux audio d'entrée du ADC audio 142.

Le ADC de communication 146 échantillonne un signal analogique d'une ligne téléphonique et fournit des échantillons numérisés au processeur multimédia 110. La manière dont le processeur multimédia 110 traite les échantillons numérisés dépend de la fonction mise en oeuvre dans le micrologiciel. Par exemple, le processeur multimédia 110 peut mettre en oeuvre des fonctions modem en exécutant des programmes dans le micrologiciel qui accomplit une démodulation V.34 des échantillons et une correction et une décompression d'erreur V.42 bis. Le processeur multimédia 110 peut également comprimer des données reçues de l'ordinateur hôte et produire des échantillons numériques représentant un signal analogique modulé correctement pour transmission par le DAC de communications 148. De façon similaire, le processeur multimédia 110 peut mettre en oeuvre des fonctions de machine de réponse, de télécopie ou vidéophone utilisant le même matériel (ADC 146 et DAC 148) que l'interface avec des lignes téléphoniques si un micrologiciel ou un logiciel convenable est disponible.

La figure 2 représente un schéma blocs d'un mode de réalisation du processeur multimédia 110. Le processeur multimédia 110 comprend un noyau de traitement 200 qui contient un processus à tout usage 210 et un processeur de vecteurs 220. Comme ici utilisé, le terme processeur

de vecteurs se réfère à un processeur qui exécute des instructions ayant des opérandes de vecteurs, c'est-à-dire des opérandes contenant chacun des éléments de données multiples du même type. Chaque processeur à tout usage 210 et processeur de vecteur 220 exécute un chemin de programme séparé et peut être un processeur scalaire ou super scalaire.

Dans le mode de réalisation à titre d'exemple, le processeur à tout usage 210 est un processeur RISC 32 bits qui fonctionne à 40 Mhz et se conforme au jeu d'instructions standard ARM7. L'architecture pour un processeur RISC ARM7 et le jeu d'instructions ARM7 sont décrits dans le ARM7DM Data Sheet disponible chez Advanced RISC Machines Ltd. Le processeur à tout usage 210 met en oeuvre également une extension du jeu d'instructions ARM7 qui comprend des instructions pour une interface au processeur de vecteurs 220. La demande de brevet en instance, ayant pour titre "System and Method for Handling Software Interrupts with Argument Passing" qui a été incorporée en référence ci-dessus décrit l'extension au jeu d'instructions ARM7 pour le mode de réalisation à titre d'exemple de l'invention. Le processeur à tout usage 210 est relié au processeur de vecteurs 220 par le bus de commande 212 pour accomplir l'extension du jeu d'instructions ARM7. De plus, une ligne d'interruption 222 est utilisée par le processeur de vecteurs 220 pour requérir une interruption sur le processeur à tout usage 210.

Le processeur de vecteurs 220 a une architecture SIMD (données multiples à instruction unique) et traite à la fois des quantités scalaires et vectorielles. Dans le mode de réalisation à titre d'exemple, le processeur de vecteurs 220 consiste en une unité de traitement centrale RISC en file d'attente qui fonctionne à 80 Mhz et a un fichier de registre de vecteurs de 288 bits. Chaque registre de vecteurs dans le fichier de registre de vecteurs peut contenir jusqu'à 32 éléments de données. Le

tableau 1 représente les types de données supportées pour des éléments de données dans un vecteur.

TABLEAU 1

Type données	Dimension donnée	Interprétation
int8	8 bits (octet)	entier complément à 2 8 bits entre -128 et 127
int9	9 bits (octet 9)	entier complément à 2/9 bits entre -256 et 255
int16	16 bits (demi-mot)	entier complément à 2 16 bits entre -32,768 et 32,767
int32	32 bits (mot)	entier complément à 2 32 bits entre -2147483648 et 2147483647
écart	32 bits (mot)	nombre points écart 32 bits se conformant au format de précision unique IEEE 754

Ainsi, un registre de vecteurs peut maintenir 32 éléments de données entier de 8 bits ou 9 bits, 16 éléments de données entier de 16 bits ou 8 éléments entier ou points d'écart 32 bits. Additionnellement, le mode de réalisation à titre d'exemple peut également fonctionner sur un opérande de vecteurs à 576 bits fractionnant deux registres de vecteurs.

Le jeu d'instructions pour le processeur de vecteurs 220 comprend des instructions pour manipuler des vecteurs et pour manipuler des scalaires. La demande de brevet ayant pour titre "Single-Instruction-Multiple-Date Processing in a Multimedia Signal Processor", qui a été incorporée par référence ci-dessus, décrit le jeu d'instructions pour le mode de réalisation à titre d'exemple de l'invention et une architecture pour mettre en oeuvre le jeu d'instructions.

Le sous-système antémémoire 230 contient un bloc SRAM 260, qui est représenté graphiquement en deux blocs,

une ROM 270 et une commande antémémoire 280. Le sous-système antémémoire 230 peut configurer le bloc SRAM 260 en (i) un cache d'instructions 262 et un cache de données 264 pour le processeur à tout usage 210 et (ii) un cache d'instructions 292 et un cache de données 294 pour le processeur de vecteurs 220. Une ROM 270 sur pastille qui contient des données et des instructions pour le processeur général 210 et le processeur de vecteurs 220 peut également être configurée en un cache. Dans le mode de réalisation à titre d'exemple, la ROM 270 contient : des procédures de réinitialisation et d'initialisation ; des procédures de diagnostics d'auto-test ; des modules logiciel d'interruption et d'exception ; et des sous routines pour une émulation "soundblaster" (programmation de son) ; des sous routines pour traitement de signal de modem V.34 ; des fonctions de téléphonie générale ; des bibliothèques de sous routines graphiques à 2 dimensions et 3 dimensions ; et des bibliothèques de sous routines pour des standards audio et vidéo tels que MEPG-1, MEPG-2, H.261, H.263, G.728 et G.723.

La figure 3 illustre les relations entre le matériel et le logiciel ou le micrologiciel dans une application de la carte multimédia 100 dans un système d'ordinateur hôte 300. Le système d'ordinateur hôte 300 a un processeur primaire 310 qui exécute des programmes mémorisés dans une mémoire principale 320. Dans le mode de réalisation à titre d'exemple, le système d'ordinateur hôte 300 est un ordinateur domestique compatible IBM coprenant un microprocesseur du type x86 et les programmes exécutés par le système d'ordinateur hôte 300 comprennent un programme d'application 330, tournant sous un système de fonctionnement tel que Windows™ 95 ou NT. Le programme d'application 330 peut communiquer avec la carte multimédia 100 par l'intermédiaire de logiciels de pilotage du dispositif 342. Les logiciels de pilotage du dispositif 342 se conforment au logiciel de pilotage du dispositif API du système de fonctionnement.

Les logiciels de pilotage ou pilotes de dispositifs sont typiquement pourvus de chaque carte multimédia 100 puisque des modes de réalisation différents de la carte multimédia 100 peuvent avoir des implantations différentes de matériel telles que des topographies de registres différents et des niveaux d'interruption différents. Le logiciel de pilotage de dispositif doit transformer correctement les signaux de commande nécessités par le mode de réalisation particulier de la carte multimédia 100 en signaux de commande tels que définis par le logiciel de pilotage de dispositif API du système de fonctionnement. Typiquement, le système de fonctionnement s'attendra à un logiciel de pilotage de dispositif différent pour chaque dispositif tel qu'un logiciel de pilotage de modem, un logiciel de pilotage graphique et un logiciel de pilotage audio. Ainsi, si un mode de réalisation de la carte multimédia 100 combine la fonctionnalité d'une carte audio, d'un modem et une carte graphique, trois logiciels de pilotage de dispositifs séparés sont typiquement exigés par le système de fonctionnement.

Le processeur à tout usage 210 dans le processeur multimédia 110 exécute un système de fonctionnement en temps réel 360 qui contrôle des communications avec les logiciels de pilotage de dispositifs 342. Le processeur à tout usage 210 accomplit également des tâches générales 370. Le processeur de vecteurs 220 accomplit des tâches vectorielles 380.

Le sous-système antémémoire 230 (figure 2) couple également les processeurs généraux 210 et les processeurs de vecteurs 220 à deux bus système : IOBUS 240 et FBUS 250. IOBUS 240 fonctionne typiquement à une fréquence inférieure à FBUS 250. Des dispositifs à vitesse plus lente sont couplés au IOBUS 240, tandis que des dispositifs à vitesse plus élevée sont couplés au FBUS 250. En séparant les dispositifs à vitesse plus lente des dispositifs à vitesse plus élevée, les dispositifs à

vitesse plus lente sont empêchés d'avoir indûment un impact sur la performance des dispositifs à vitesse plus élevée.

Le sous-système antémémoire 230 sert également comme commutateur pour une communication entre le IOBUS 240, le FBUS 250, le processeur général 210 et le processeur de vecteurs 220. Dans la plupart des modes de réalisation du sous-système antémémoire 230, des accès simultanés multiples entre les bus et les processeurs sont possibles. Par exemple, le processeur de vecteurs 220 est capable de communiquer avec le FBUS 250 en même temps que le processeur à tout usage 210 communique avec le IOBUS 240. Dans un mode de réalisation de l'invention, la combinaison du commutateur et de la fonction cache est accomplie en utilisant des techniques de topographie directe pour le FBUS 250 et le IOBUS 240. Spécifiquement, les dispositifs sur le FBUS 250 et le IOBUS 240 peuvent être accédés par le processeur à tout usage 210 et le processeur de vecteurs 220 par des lectures et écritures de mémoire standards à des adresses appropriées.

La figure 5 représente la topographie de mémoire d'un mode de réalisation de l'invention. Le bloc mémoire 510, c'est-à-dire l'espace d'adresse du zéro d'adresse d'octet à la 4M - 1 adresse (les unités M et V, qui sont utilisées dans la présente description comme unités pour des adresses de mémoire, représentent les nombres 1 048 576 (c'est-à-dire 1024 x 1024) et 1 073 741 824 (c'est-à-dire 1024 x 1024 x 1024)), est occupé par la ROM 270. Le bloc mémoire 520, c'est-à-dire l'espace d'adresse de l'adresse d'octet 4M à 8M - 1, est occupé par le bloc SRAM 260. Le bloc mémoire 530, c'est-à-dire l'espace d'adresse de l'adresse d'octet 8M à l'adresse d'octet 72M - 1, est occupé par la mémoire locale 120. Les dispositifs sur le FBUS 250 sont topographiés au bloc mémoire 540 qui démarre après l'adresse d'octet 72M et s'étend à l'adresse d'octet 77M. Le bloc mémoire 550 est réservé pour l'expansion future. Les dispositifs sur le

IOBUS 240 sont topographiés au bloc mémoire 560, qui démarre après l'adresse d'octet 125M et s'étend à l'adresse d'octet 128M - 1. Le bloc mémoire 570 est également réservé pour une expansion future. Le bloc mémoire 580, c'est-à-dire l'espace d'adresses de l'adresse d'octet 2G à l'adresse d'octet 4G-1, est occupé par d'autres dispositifs d'ordinateur hôte et est typiquement accédé par l'interface de bus local 255.

Le FBUS 250 (figure 2) est relié à un contrôleur de mémoire 258, un contrôleur de DMA 257, une interface de bus local 255 et une interface de dispositif 252 qui fournissent des interfaces respectivement pour la mémoire locale 120, le bus local 105 et des convertisseurs 132, 134, 142, 144, 146, 148 et 150 représentés en figure 1.

Le contrôleur de mémoire 258 contrôle des lectures et écritures à la mémoire locale 120. Dans le mode de réalisation à titre d'exemple, le contrôleur de mémoire 258 contrôle une banque de DRAMs (deux pastilles SRAM 1Mx16) configurée pour utiliser 24 à 26 bits d'adresses et 32 bits de données et ayant les caractéristiques : (i) d'un protocole de rafraîchissement "CAS-avant-RAS", accompli à une fréquence de rafraîchissement programmable, (ii) d'écritures partielles qui initialisent des opérations de lecture-modification-écriture et (iii) d'entrelacement de banques internes. Le contrôleur de mémoire 258 fournit également un accord de fréquence de 1:1 entre la mémoire 120 et le FBUS 250, "à la fois une précharge de banque" manuelle et une mise en file d'attente d'adresses et de données pour mieux utiliser le FBUS 250. Les DRAMs synchrones sont connues pour efficacement fonctionner à de telles fréquences (80 MHz) et des DRAMs de page virtuelle rapide standard et des DRAMs de données étendues (EDO) pourraient être également utilisées. Des contrôleurs de DRAM à capacités similaires au contrôleur de mémoire 258 dans le mode de réalisation à titre d'exemple sont connus dans l'art.

Le contrôleur de DMA 257 contrôle des accès directs en mémoire entre la mémoire principale de l'ordinateur hôte et la mémoire locale du processeur multimédia 200. De tels contrôleurs DMA sont bien connus dans l'art. Dans certains modes de réalisation de l'invention, un transfert de données de mémoire est inclus. Le transfert de données de mémoire accomplit un accès direct en mémoire d'un bloc de mémoire à un autre bloc de mémoire.

L'interface de bus local 255 met en oeuvre le protocole exigé pour des communications avec l'ordinateur hôte par l'intermédiaire du bus local 105. Dans le mode de réalisation à titre d'exemple, l'interface de bus local 255 réalise une interface à un bus PCI de 32 bits, 33 MHz. De telles interfaces sont bien connues dans l'art.

L'interface de dispositif 252 fournit une interface de matériel pour des dispositifs tels que les convertisseurs 132, 134, 142, 144, 146, 148 et 150 qui seraient typiquement sur une plaque à circuit imprimé avec le processeur multimédia 110. L'interface de dispositif 252 peut être personnalisée pour l'application particulière du processeur multimédia 110. En particulier, l'interface de dispositif 252 pourrait seulement réaliser une interface pour des dispositifs spécifiques ou des circuits intégrés. Des unités typiques dans l'interface du dispositif 252 réalisent une interface pour connexion de ADCs, DACs ou CODECs standards. Les conceptions pour des interfaces ADC, DAC et CODEC sont bien connues dans l'art et non davantage décrits ici. D'autres interfaces qui pourraient utilisées comprennent une interface ISDN pour téléphone numérique et des interfaces pour des bus tels que pour un bus de microcanaux, mais ne sont pas limitées à celles-ci. Dans un mode de réalisation du processeur multimédia 110, l'interface du dispositif 252 est un ASIC (circuit intégré d'application sépcifique) qui peut être programmé pour accomplir une fonctionnalité souhaitée.

Le IOBUS 240 fonctionne à une fréquence (40 MHz) qui est inférieure à la fréquence de fonctionnement (80 MHz) du bus 250. Couplés au IOBUS 240 sont une temporisation de système 242, un UART 243 (interface de transmission asynchrone), un processeur de flux de bits 248 et un contrôleur d'interruption 245. La temporisation de système 242 interrompt le processeur 210 à des intervalles planifiés qui sont sélectionnés en écrivant à des registres correspondant à la temporisation de système 242. Dans le mode de réalisation à titre d'exemple, la temporisation de système 242 est un Intel 8254 standard compatible à la temporisation d'intervalle ayant trois compteurs indépendants de 16 bits et six modes de compteurs programmables.

L'UART 243 est une interface série, qui est compatible avec le circuit intégré UART 16450 très connu, pour utilisation dans un modem ou des applications de télécopie qui exige un port de communication série standard ("COM") d'un ordinateur domestique.

Le processeur de flux de bits 245 est un processeur de matériel fixe qui accomplit les fonctions spécifiques sur un flux de bits d'entrée ou de sortie. Dans le mode de réalisation à titre d'exemple, le processeur de flux de bits 245 accomplit des étages initiaux ou finaux de codage ou de décodage MPEG. En particulier, le processeur de flux de bits 245 accomplit un codage et un décodage de longueur variable (Huffman) et comprime et décomprime des données vidéo en format en "zig-zag". Le processeur de flux de bits 245 fonctionne en parallèle avec le processeur à tout usage 210 et le processeur de vecteurs 220 et sous le contrôle de ceux-ci. Les processeurs 210 et 220 configurent le processeur de flux de bits 245 par l'intermédiaire de registres de contrôle. La demande de brevet US en instance ayant pour titre "Methods and Apparatus for Processing Video Data", qui a été incorporée à titre de référence ci-dessus, décrit un mode

de réalisation à titre d'exemple du processeur à flux de bits 245.

Le contrôleur d'interruption 248 commande des interruptions du processeur à tout usage 210 et supporte des priorités d'interruption multiples. Un registre de masquage est prévu pour permettre à chaque priorité d'interruption d'être masquée individuellement. Dans le mode de réalisation à titre d'exemple, le controleur d'interruption 245 est programmable et met en oeuvre le système d'interruption standard Intel 8259 qui est commun aux ordinateurs domestiques à base du x86. Une interruption de priorité la plus élevée (niveau 0) est attribuée à la temporisation du système 242. Des niveaux de priorité 1, 2, 3 et 7 sont respectivement attribués à une mémoire tampon d'image virtuelle, un contrôleur de DMA 257 et une interface de dispositif 252, un processeur de flux de bits 245, une interface de bus local 255 et un UART 243. Des niveaux de priorité d'interruption 4, 5 et 6 ne sont pas attribués dans le mode de réalisation à titre d'exemple de l'invention. La mémoire tampon d'image virtuelle au niveau de priorité 1, qui est incluse dans certains modes de réalisation de l'invention, simule une mémoire tampon d'image VGA standard.

La figure 4 représente un schéma blocs du sous-système antémémoire 230. Le bloc SRAM 260 est divisé en quatre banques de mémoire pour former un cache d'instructions 262 et un cache de données 264 pour utilisation avec le processeur général 210, ainsi qu'un cache d'instructions 292 et un cache de données 294 pour utilisation avec le processeur de vecteurs 220. Le bloc SRAM 260 contient également une section de référence 406, qui est sous divisée pour chacune des banques de mémoire. Le bloc SRAM 260 est un circuit de mémoire à double port avec un port de lecture 440 et un port d'écriture 430, de sorte qu'une lecture et une écriture simultanées du bloc SRAM 260 est supportée. Le sous-système antémémoire 230 contient également un cache de ROM 270, ayant une image

de référence 472. Comme expliqué ci-dessus le cache de ROM 270 contient des instructions fréquemment utilisées et des données pour le processeur général 210 et le processeur de vecteurs 220. Bien que l'image de référence 472 ne puisse pas être modifiée, des adresses individuelles peuvent être marquées comme invalides de sorte que des données ou des instructions peuvent être amenées de la mémoire à utiliser à la place de données ou des instructions dans la ROM 270.

Une file d'attente de données 410 accomplit la fonction de commutateur de données du sous-système antémémoire 230. La file d'attente de données 410 est capable de créer des trajets de communications de données simultanées multiples entre le IOBUS 240, le FBUS 250, le processeur à tout usage 210, le processeur de vecteur 220 et le bloc SRAM 260. De façon similaire, la file d'attente d'adresses 420 accomplit des fonctions de commutation pour des adresses. Dans le mode de réalisation de la figure 4, le IOBUS 240 et le FBUS 250 utilisent un multiplexage temporel pour des signaux d'adresses de données. La commande antémémoire 280 fournit les lignes de commande à la file d'attente de données 410 et à la file d'attente d'adresses 420 pour configurer correctement les canaux de communication.

Dans certains modes de réalisation de ce système antémémoire 230, un protocole à base de transactions est utilisé pour supporter toutes les opérations de lecture et d'écriture. Toute unité couplée à ce système antémémoire 230, telle que le processeur général 210, le processeur de vecteurs 220 ou les divers dispositifs sur le IOBUS 240 et le FBUS 250, peut placer une requête au sous-système antémémoire 230. Une telle requête est formée par un code d'identification de dispositif ("dispositif ID") et une adresse de l'emplacement de mémoire requis. Chaque unité a un dispositif ID distinct et le sous-système antémémoire 230 peut privilégier les requêtes basées sur le dispositif ID de l'unité réalisant

la requête. Lorsque les données à l'adresse requise deviennent disponibles, le sous-système antémémoire répond au dispositif ID, à un code d'identification de transaction ("transaction ID"), à l'adresse et aux données requises. Si l'adresse requise n'est pas contenue dans le bloc SRAM 260 ou la ROM 270, le sous-système antémémoire 230 ne sera pas capable de répondre à la requête spécifique pendant plusieurs cycles d'horloge alors que les données à l'adresse de mémoire sont retirées. Cependant, alors que les données d'une première requête sont en train d'être retirées, le sous-système antémémoire 230 est capable de traiter une seconde requête d'une unité différente avec un dispositif ID différent. De cette manière, la requête en instance ne bloquera pas des requêtes subséquentes d'autres unités. De plus, le sous-système antémémoire 230 peut supporter une requête de lecture et une requête d'écriture simultanément dans un cycle unique.

Comme expliqué ci-dessus, le bloc SRAM 260 est divisé en quatre banques de mémoire. Le bloc SRAM 260 est à double port, ayant un port de lecture 440 et un port d'écriture 430, de sorte que dans n'importe quel cycle, le bloc SRAM 260 peut accepter une requête de lecture et une requête d'écriture. La section TAG (référence) 406 du bloc SRAM 260 doit avoir deux ports de lecture pour supporter les requêtes de lecture et d'écriture simultanées. Ainsi, l'adresse utilisée par le port de lecture 440 ainsi que l'adresse utilisée par le port d'écriture 430 peuvent être comparées à des références cache pour réussir ou rater des conditions simultanément. La section de référence 406 contient également un port d'écriture de sorte que comme la requête d'écriture au port d'écriture 430 est accomplie, les zones de référence appropriées sont également changées.

Dépendant des contraintes du système global, le sous-système antémémoire 230 peut être utilisé avec des politiques de cache soit de réponse soit de

surimpression. De plus, dans certains modes de réalisation, pour augmenter davantage la vitesse, la dimension de lignes de cache peut être rendue deux fois la largeur de données. Dans ces modes de réalisation, pour des buts de "conservation de dossier", on doit attribuer à chaque ligne de cache deux bits valides et deux bits modifiés, puisque chaque ligne de cache contient deux vecteurs. Le bloc SRAM 260 doit également effacer globalement tous les bits valides si un signal d'effacement global est reçu. Dans d'autres modes de réalisation, des signaux d'effacement individuels sont supportés pour chaque banque dans le bloc SRAM 260.

La figure 6 est un schéma blocs d'un mode de réalisation d'une file d'attente de données 410. Puisque le sous-système antémémoire 230 est à la fois un système antémémoire et un commutateur pour le IOBUS 240, le FBUS 250, le processeur à tout usage 210 et le processeur de vecteurs 220, les bus et le processeur doivent être capables de communiquer soit à travers le cache soit directement si le cache est en train d'être utilisé par un autre dispositif. Les processeurs sont généralement plus rapides que les dispositifs sur les bus ; de ce fait, les processeurs utiliseront généralement le cache sur des écritures et permettront au système de retour d'écriture de cache de placer des données au dispositif de bus approprié. De façon similaire, les processeurs requièrent généralement des informations du cache plutôt que des dispositifs directement. Si le cache ne contient pas les données requises, les processeurs reposent typiquement sur le sous-système antémémoire pour retirer les données requises dans le cache et produire les données aux processeurs. Cependant, lorsque le cache est occupé les processeurs peuvent accéder directement aux bus.

Les données sont transférées du processeur à tout usage 210 au IOBUS 240 par un multiplexeur entrée/sortie MUX IO 630. Des données du IOBUS 240 au processeur à tout

usage 210 passent à travers le multiplexeur MUX 620 de lecture universelle. Des données sont transférées soit du bloc SRAM 260 soit de la ROM 207 au processeur à tout usage 210 à travers le multiplexeur MUX 650 de lecture de cache et le multiplexeur MUX 620 de lecture universelle. Des données sont transférées du processeur universel 210 au bloc SRAM 260 par le multiplexeur MUX 610 d'écriture de cache. Le multiplexeur MUX 650 de lecture de cache, le multiplexeur MUX 610 d'écriture de cache, le multiplexeur 630 IO (entrée/sortie) et le multiplexeur MUX 620 de lecture universelle peuvent être des multiplexeurs conventionnels et peuvent contenir des circuits de verrouillage internes ou des registres comme nécessaire pour des contraintes de temporisation. Les lignes de commande de sélection (non représentées) des multiplexeurs sont dictées par une commande de cache 280 (figure 4). Des données sont transférées du processeur universel 210 au FBUS 250 par le multiplexeur MUX 610 à écriture de cache et le multiplexeur MUX 640 du FBUS. Des données du FBUS 250 au processeur universel 210 sont analysées par un circuit ou mémoire tampon 660, le multiplexeur MUX 650 de lecture de cache et le multiplexeur MUX 620 de lecture universelle. Pour accomplir ces fonctions, la mémoire tampon 660 peut être une mémoire tampon, un circuit de verrouillage ou un registre conventionnel.

Le processeur à tout usage 210 peut commander le processeur de vecteurs 220 par des lignes de commande 212 (figure 2). Un transfert de données direct entre le processeur à tout usage 210 et le processeur de vecteurs 220 n'est généralement pas requis mais peut être accompli par le bloc SRAM 260 ou n'importe quel autre dispositif puisque les deux processeurs partagent une topographie de mémoire commune.

Des données de la ROM 270 et du bloc SRAM 260 au IOBUS 240 circulent à travers le multiplexeur de lecture de cache 650 et le multiplexeur d'entrée/sortie 630. Des

données du IOBUS 240 au bloc SRAM 260 circulent à travers le multiplexeur d'écriture de cache 610. Des données du IOBUS 240 au FBUS 250 passent à travers le multiplexeur d'écriture de cache 610 et le multiplexeur FBUS 640. Des données pour le IOBUS 240 du FBUS 250 passent à travers la mémoire tampon 660, le multiplexeur de lecture de cache 650 et le multiplexeur d'entrée/sortie 630. Des données pour l'IOBUS 240 du processeur de vecteurs 220 passent à travers le multiplexeur d'écriture de cache 610 et le multiplexeur d'entrée/sortie 630. Des données du IOBUS 240 au processeur de vecteurs 220 passent à travers le multiplexeur de lecture de cache 650. Dans certains modes de réalisation de l'invention, la file d'attente directe pour des données du processeur de vecteurs 220 au IOBUS 240 est éliminée pour simplifier la conception de la file d'attente de données 410. Puisque la largeur de bande du processeur de vecteurs 220 est beaucoup plus grande que la largeur de bande du IOBUS 240, un trajet de communication direct du processeur de vecteurs 220 au IOBUS 240 devient très inefficace par rapport au temps de traitement du processeur de vecteurs 220.

Des données pour le FBUS 250 au bloc SRAM 260 et la ROM 270 passent à travers le multiplexeur de lecture de cache 650 et le multiplexeur du FBUS 640. Des données du FBUS 250 au bloc SRAM 260 passent à travers la mémoire tampon 660 et le multiplexeur d'écriture de cache 610. Des données du FBUS 250 peuvent directement atteindre le processeur de vecteurs 220 par la mémoire tampon 660 et le multiplexeur de lecture de cache 650. Des données pour le FBUS 250 peuvent également venir directement du processeur de vecteurs 220 à travers le multiplexeur d'écriture de cache 610 et le multiplexeur FBUS 640.

Des données du processeur de vecteurs 220 circulent au bloc SRAM 260 à travers le multiplexeur d'écriture de cache 610. Des données du bloc SRAM 260 et de la ROM 270 passent à travers le multiplexeur de lecture de cache 650 au processeur de vecteurs 220.

La figure 7 est un schéma blocs détaillé d'un second mode de réalisation de la file d'attente de données 410. Puisque la fonctionnalité du mode de réalisation de la figure 7 est similaire à la fonctionnalité du mode de réalisation de la figure 6, seulement les différences entre les modes de réalisation sont discutées en détail. Cependant, les organisations générales des éléments dans chaque mode de réalisation sont également décrites. En figure 7, le multiplexeur de lecture de cache 650 est remplacé par le multiplexeur de lecture de cache 750 et le verrouilleur multiplexeur 751. La mémoire tampon 660 est remplacée par le verrouilleur de lecture 760. Le multiplexeur FBUS 640 est remplacé par le multiplexeur FBUS 640, le verrouilleur de données d'écriture (WB) 741, le verrouilleur d'écriture de mémoire 742 et le verrouilleur d'écriture de mémoire 743. Les verrouilleurs dans le mode de réalisation de la figure 7 sont utilisés pour mettre en file d'attente la fil d'attente de données. Le multiplexeur d'écriture de cache 610 est remplacé par le multiplexeur d'écriture de cache 710, le verrouilleur de données d'écriture 712, un dispositif aligneur 713 et un verrouilleur d'écriture d'entrée/sortie 711. Le multiplexeur d'entrée/sortie 630 est remplacé par le verrouilleur de lecture d'entrée/sortie 731 et le verrouilleur de lecture d'entrée/sortie 732. Le multiplexeur de lecture universel 620 est remplacé par le verrouilleur d'écriture d'entrée/sortie 721 et le micro antémémoire 722.

Le micro cache 722 couple l'antémémoire principal, le bloc SRAM 270 et la ROM 260 au processeur à tout usage 210. Le micro cache 722 est divisé en un micro cache d'instructions et un micro cache de données, dont chacun comprend une portion de référence 822 (figure 8), des comparateurs de référence et des bits valides. Le micro cache 722 fonctionne comme une mémoire tampon de prélecture. L'adresse d'une requête du processeur à tout usage 210 est tout d'abord comparée à la portion de

référence 822 du micro cache 722. Si un manquement du micro cache se produit (c'est-à-dire aucun accord dans la référence de micro cache 822) l'adresse de la requête avec l'adresse et autres informations de commande est transmise à l'antémémoire principal. Pour simplifier le micro cache 722, des écritures de données du processeur à tout usage 210 qui accorde une référence dans le micro cache 722 invalident l'adresse de micro cache de sorte que les données écrites doivent être transmises au micro cache principal. De cette manière, une cohérence de caches peut être maintenue sans conceptions pour réécriture et sur écriture complexes sur le micro cache 722.

La figure 8 représente un schéma blocs d'un mode de réalisation d'une file d'attente d'adresses 420. L'interface FBUS est composée d'une file d'attente d'adresses à quatre entrées et d'un verrouilleur de pré-écriture. L'interface FBUS 850 peut simultanément supporter une lecture en instance de l'antémémoire d'instructions 262, une lecture en instance de l'antémémoire d'instructions 292, une requête d'écriture de l'antémémoire de données 294 et une requête de ré-écriture de l'antémémoire de données 294. Les adresses pour les requêtes d'écriture sont transmises au multiplexeur d'adresses d'écriture 210 tandis que les adresses pour les requêtes de lecture sont transmises au multiplexeur d'adresses de lecture 820. Un contrôleur d'antémémoire 280 (figure 2) accomplit un arbitrage entre des requêtes du processeur à tout usage 210, du processeur de vecteurs 220, du IOBUS 240 et du FBUS 250 sur la base du dispositif ID de la requête. Le contrôleur d'antémémoire 280 configure alors les diverses multiplexeurs de la file d'attente de données 410 et de la file d'attente de données 420 pour maintenir les requêtes. Le schéma d'arbitrage peut être décidé sur la base d'une estimation de l'importance de chaque dispositif. De façon typique, on donne au processeur à

tout usage 210 la priorité la plus élevée. Comme expliqué ci-dessus, le sous-système antémémoire 230 est capable d'opérations de lecture et d'écriture simultanées. De ce fait, des comparateurs séparés sont nécessaires pour les requêtes de lecture et d'écriture. Le comparateur 811 est utilisé pour comparer l'adresse d'écriture du multiplexeur d'adresses d'écriture 810 aux adresses reçues par le port de référence d'écriture 406-1 pour déterminer si l'adresse d'écriture de la requête actuelle se trouve dans l'antémémoire. Si l'adresse est dans l'antémémoire, l'antémémoire est remis à jour par les nouvelles données à l'emplacement d'antémémoire d'accord. Si l'adresse n'est pas dans l'antémémoire, l'adresse et les données sont écrites à l'antémémoire dans un emplacement d'antémémoire non utilisé où l'emplacement qui est le moins accédé récemment. Eventuellement, les données sont transmises au dispositif cartographié adressé direct correct en utilisant des techniques d'antémémoire à ré-écriture ou surécriture.

Le comparateur 821 est utilisé pour comparer l'adresse de lecture de requête de lecture du multiplexeur d'adresses de lecture 820 et les adresses reçues par le port de référence de lecture 406-2. Si une référence concorde avec l'adresse de lecture, les données associées à la référence sont transmises au dispositif de requête en utilisant la file d'attente de données 410. Comme expliqué ci-dessus, si un protocole de transaction est utilisé, les données seront retournées à un dispositif ID, une transaction ID et l'adresse requise. Si aucune référence ne concorde avec l'adresse de lecture, le sous-système antémémoire 230 doit retirer les données requises du dispositif topographié à mémoire directe appropriée. Lorsque les données requises sont retirées, les données requises, le dispositif ID, la transaction ID et l'adresse sont transmises au dispositif de requête. Pendant que les données pour une première requête sont en train d'être retirées, le sous-système

antémémoire 230 est capable de servir une seconde requête de lecture de sorte qu'un second dispositif exigeant l'antémémoire n'est pas bloqué par le premier dispositif.

Les divers modes de réalisation de la structure de cette invention qui sont décrits ci-dessus sont seulement à titre d'illustration des principes de l'invention et ne sont pas destinés à limiter la portée de l'invention aux modes de réalisation particuliers décrits. Au vu de la présente description, ceux de l'art peuvent définir dans la portée de la présente invention d'autres mises en oeuvre des files d'attente de données, des commutateurs, des files d'attente d'adresses, des sous-systèmes antémémoire, des multiplexeurs, des verrouilleurs, des bus, des processeurs et utiliser ces caractéristiques alternatives pour créer un processeur de signaux numériques.

REVENDICATIONS

1. Dispositif processeur intégré de signaux numériques, caractérisé en ce qu'il comprend :

un processeur à tout usage (210) ; et

un processeur de vecteurs (220) capable de fonctionner en parallèle avec le processeur à tout usage (210).

2. Dispositif selon la revendication 1, caractérisé en ce que le processeur à tout usage (210) comprend ;

un jeu de registres scalaire ;

une unité de décodage d'instructions ; et

un noyau de traitement (200) qui traite un certain nombre de valeurs scalaires selon des instructions décodées par l'unité de décodage d'instructions.

3. Dispositif selon la revendication 2, caractérisé en ce que le processeur de vecteurs (220) comprend :

un jeu de registres de vecteurs ;

une seconde unité de décodage d'instructions et un second noyau de traitement qui traite un certain nombre de valeurs de vecteurs selon des instructions décodées par la seconde unité de décodage d'instructions.

4. Dispositif selon la revendication 1, caractérisé en ce qu'il comprend de plus un sous-système antémémoire (230) couplé au processeur à tout usage (210) et au processeur de vecteurs (220), le sous-système antémémoire (230) ayant un cache de mémoire.

5. Dispositif selon la revendication 4, caractérisé en ce que le sous-système antémémoire (230) comprend :

un port de lecture d'antémémoire ; et

un port d'écriture d'antémémoire ;

et en ce que le sous-système antémémoire (230) supporte des accès simultanés au port de lecture d'antémémoire et au port d'écriture d'antémémoire.

6. Dispositif selon la revendication 4, caractérisé en ce que le sous-système antémémoire (230) comprend :

une file d'attente de données (410) couplée au processeur à tout usage (210) et au processeur de vecteurs (220) ;

une file d'attente d'adresses (420) couplée au processeur à tout usage (210) et au processeur de vecteurs (220) et en ce que le cache de mémoire comprend un cache SRAM (260) couplé à la file d'attente de données (410) et à la file d'attente d'adresses (420) et un cache de ROM (270) couplé à la file d'attente de données (410) et à la file d'attente d'adresses (420).

7. Dispositif selon la revendication 4, caractérisé en ce qu'il comprend de plus un premier bus couplé au sous-système antémémoire (230) et un second bus couplé au sous-système antémémoire (230).

8. Dispositif selon la revendication 7, caractérisé en ce que le premier bus a une largeur de bande de premier bus supérieure à une largeur de bande de second bus du second bus.

9. Dispositif selon la revendication 8, caractérisé en ce qu'il comprend de plus un processeur de flux de bits (245) couplé au second bus et une interface de bus local couplée au premier bus.

10. Dispositif selon la revendication 9, caractérisé en ce que l'interface du bus local est couplée à un bus local (105) d'un processeur primaire d'un système d'ordinateur.

11. Dispositif selon la revendication 9, caractérisé en ce que le sous-système antémémoire (230) peut être configuré pour fournir une pluralité de trajets de communication entre le processeur de vecteurs (220), le processeur à tout usage (210), le premier bus et le second bus.

12. Dispositif selon la revendication 7, caractérisé en ce que le sous-système antémémoire (230) est accédé par une première requête de cache.

13. Dispositif selon la revendication 12, caractérisé en ce que si la première requête de cache

précitée exige à un certain nombre de cycles d'être accomplis, le sous-système antémémoire (230) est capable d'accepter une seconde requête de cache avant de finir ladite première requête de cache.

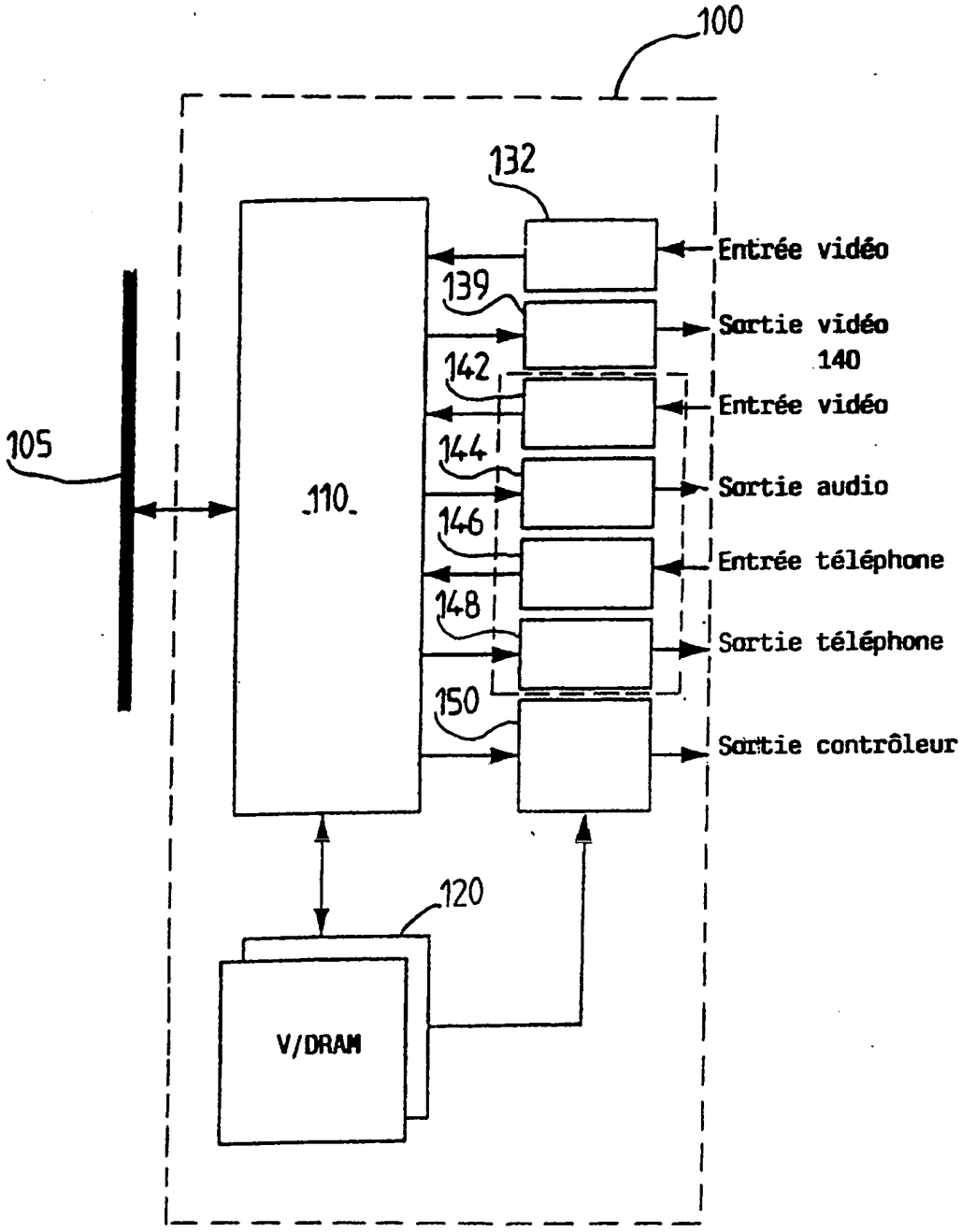

100
132
Entrée vidéo
139
Sortie vidéo
140
142
Entrée vidéo
105
144
Sortie audio
110
146
Entrée téléphone
148
Sortie téléphone
150
Sortie contrôleur
120
V/DRAM

Fig. 1

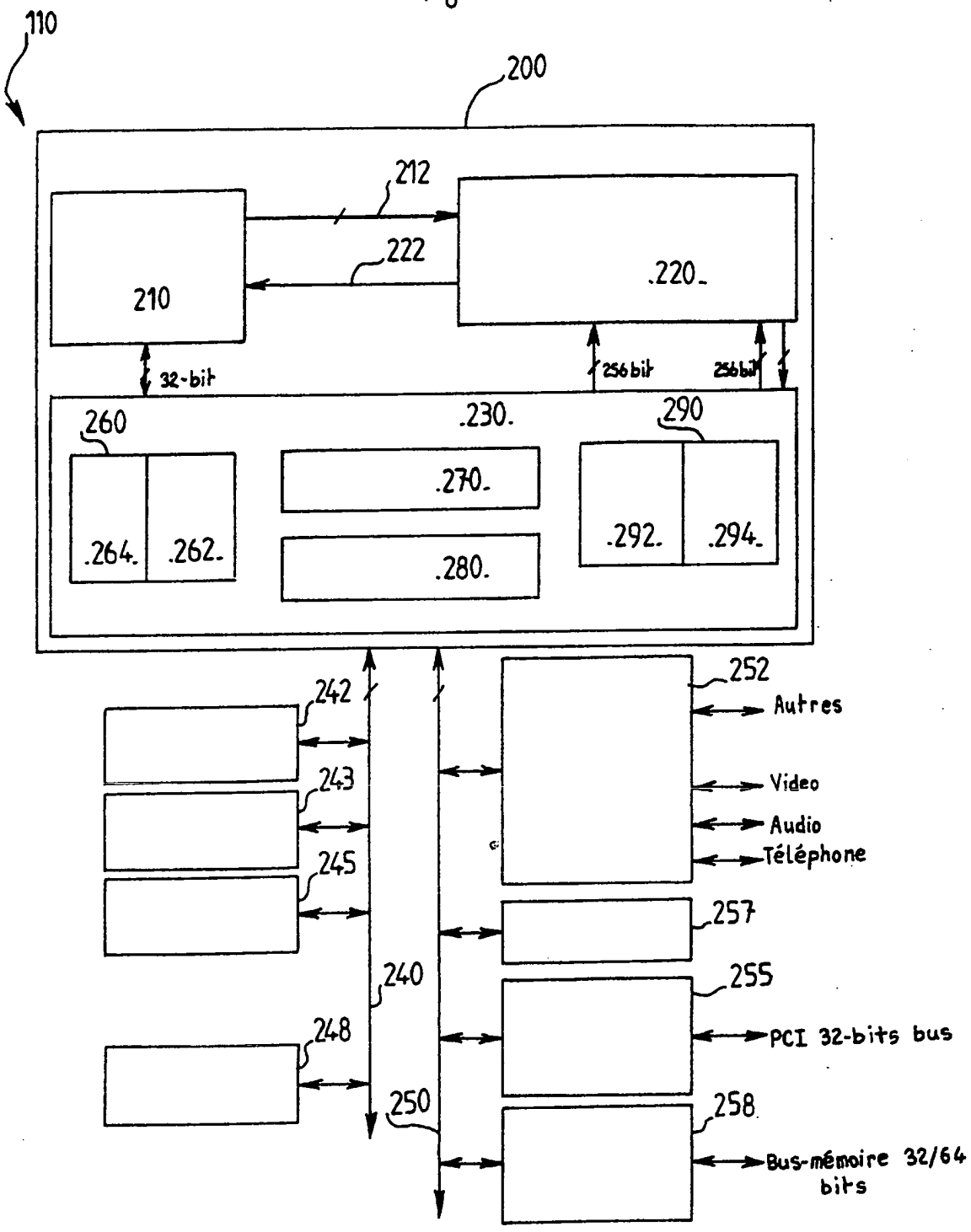

110
200
212
222
210
.220.
32-bit
256bit
256bit
260
.230.
290
.264.
.262.
.270.
.280.
.292.
.294.
242
243
245
240
248
250
252
Autres
Video
Audio
Téléphone
257
255
PCI 32-bits bus
258
Bus-mémoire 32/64 bits

Fig. 2

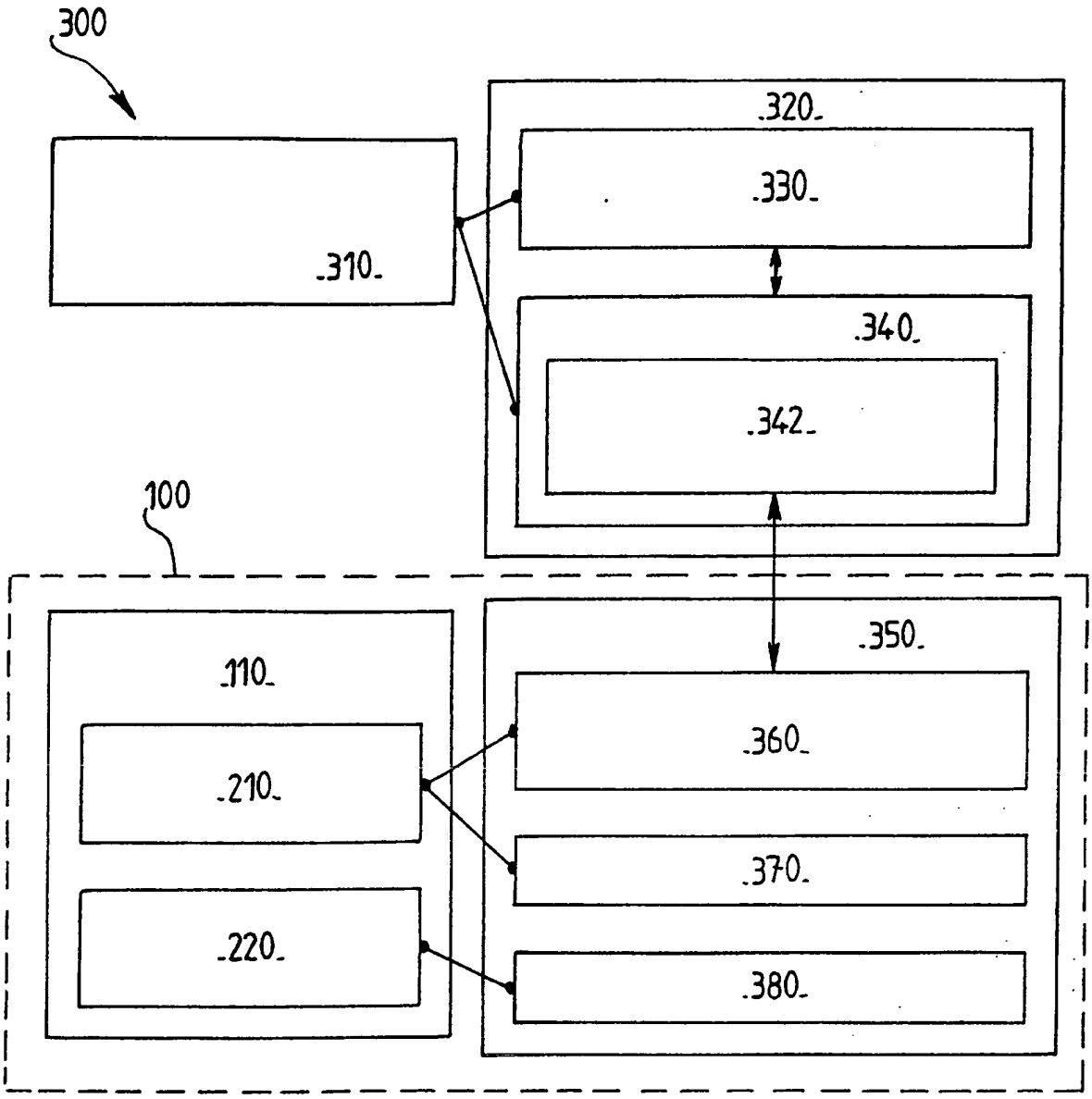


Fig 3

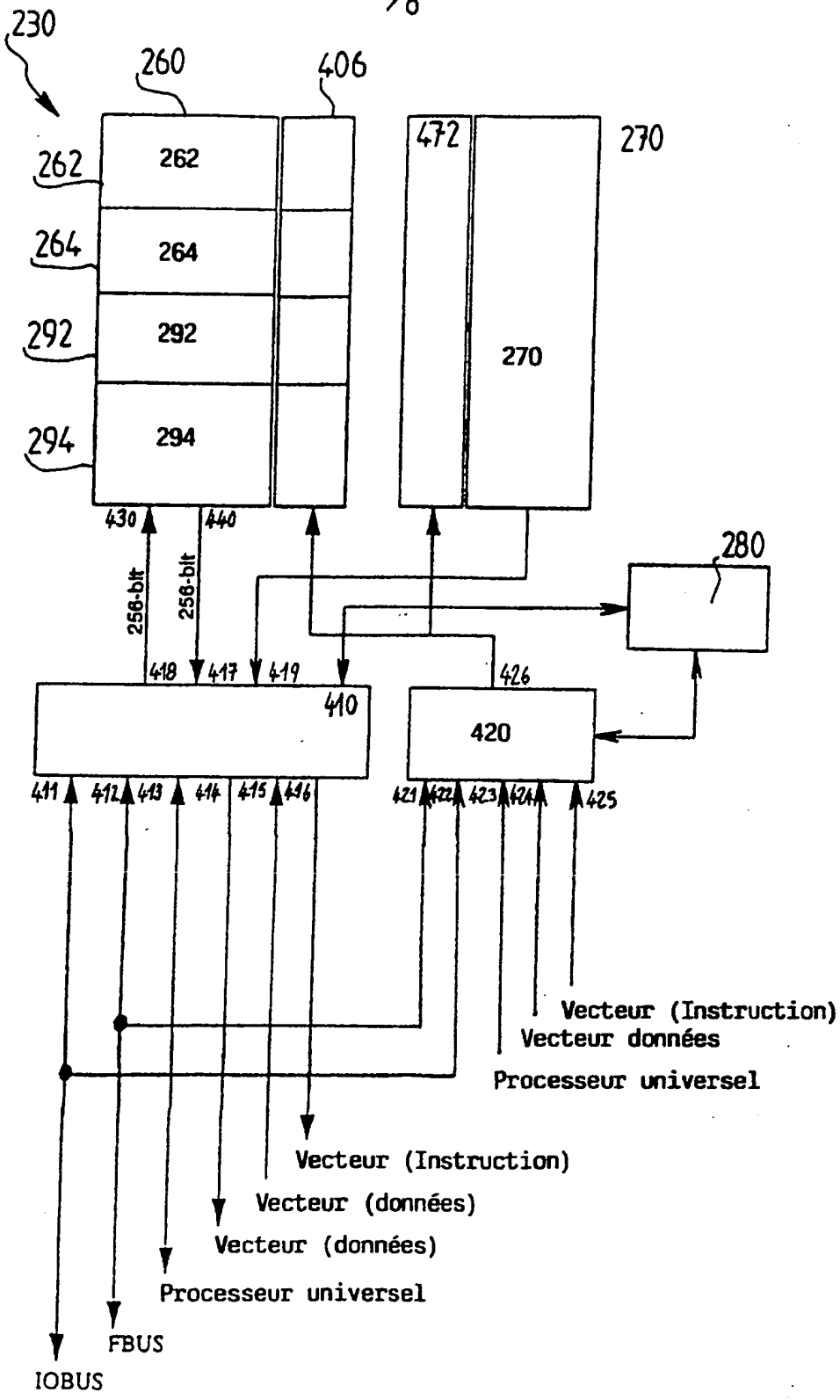

230
260
406
262
264
292
294
262
264
292
294
472
270
270
280
430
440
256-bit
256-bit
418
417
419
426
410
420
411
412
413
414
415
416
421
422
423
424
425
Vecteur (Instruction)
Vecteur données
Processeur universel
Vecteur (Instruction)
Vecteur (données)
Vecteur (données)
Processeur universel
FBUS
IOBUS

Fig. 4

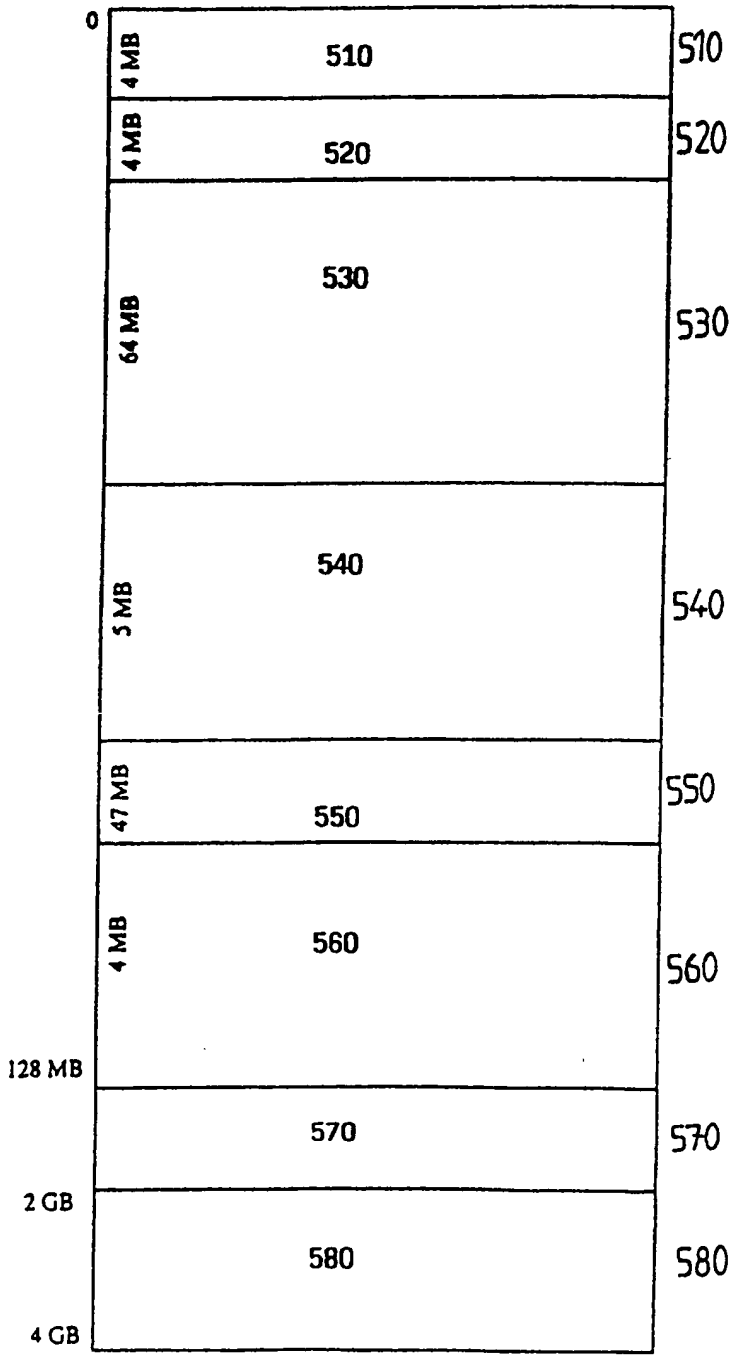

0
4 MB
510
510
4 MB
520
520
64 MB
530
530
5 MB
540
540
47 MB
550
550
4 MB
560
560
128 MB
570
570
2 GB
580
580
4 GB

Fig 5

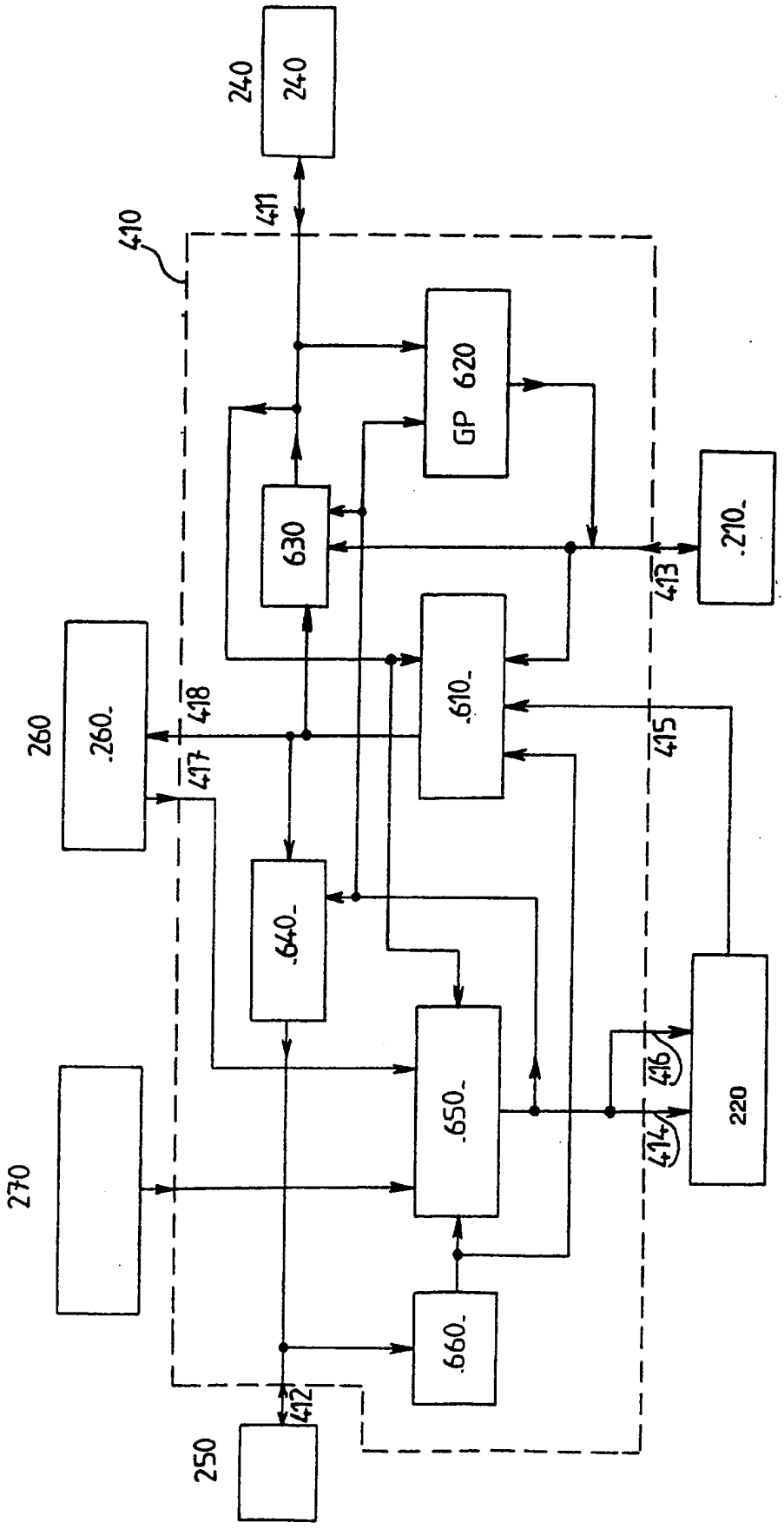

240
240
411
410
GP 620
630
.210_
413
.610_
260
.260_
418
417
415
.640_
.650_
416
414
220
270
.660_
412
250

Fig. 6

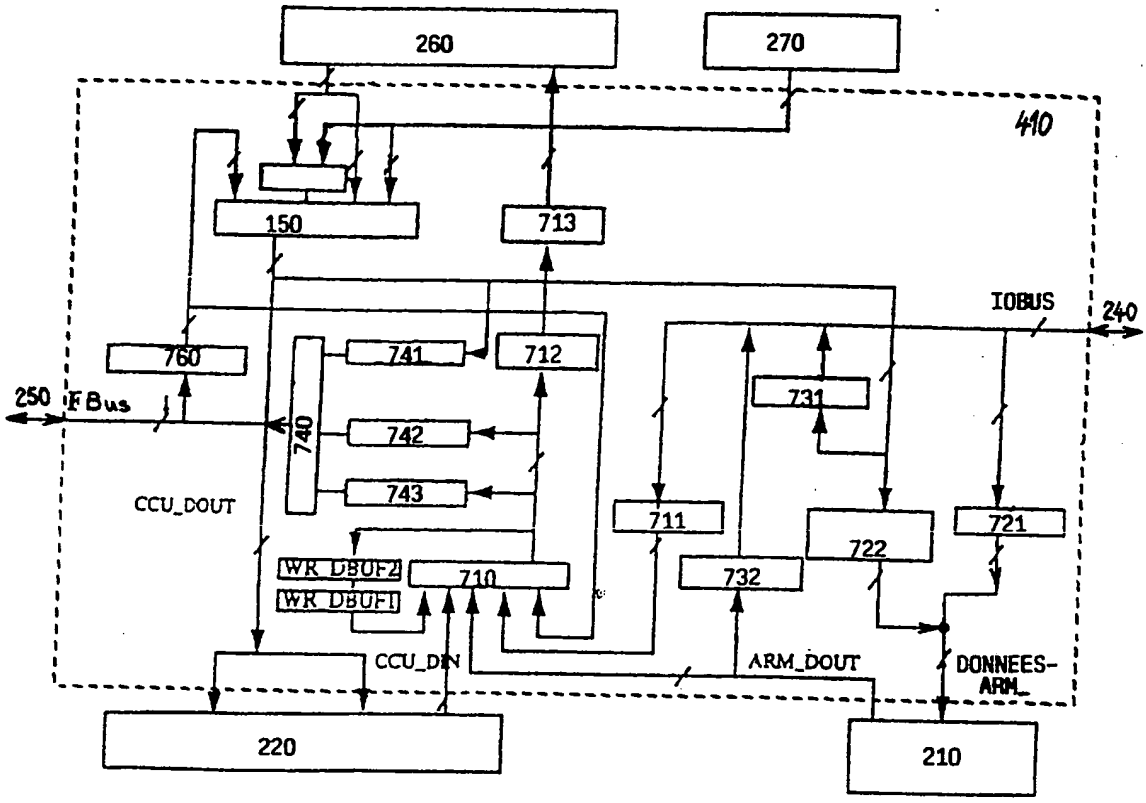

260
270
410
150
713
IOBUS
240
760
741
712
250
FBus
731
740
742
743
CCU_DOUT
711
722
721
WR_DBUF2
WR_DBUF1
710
732
CCU_DIN
ARM_DOUT
DONNEES-ARM
220
210

Fig. 7

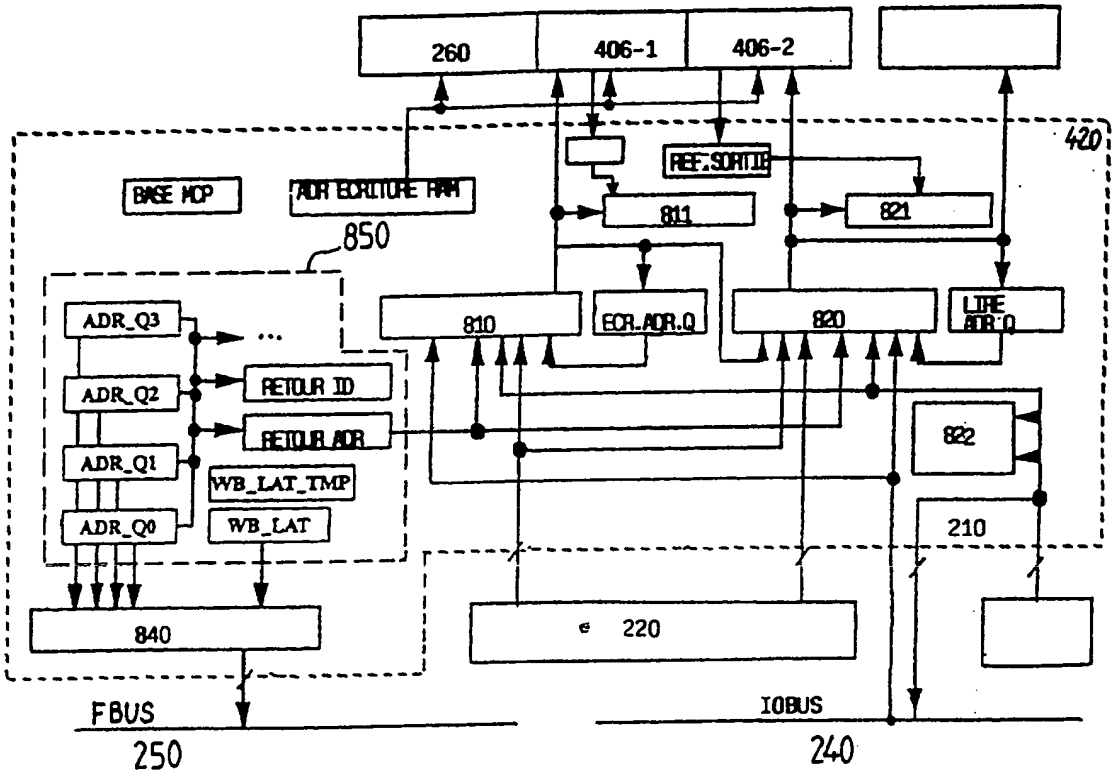

260
406-1
406-2
420
REF.SORTIE
BASE MCP
ADR ECRITURE RAM
811
821
850
ADR_Q3
...
810
ECR.ADR.Q
820
LIRE ADR.Q
ADR_Q2
RETOUR ID
RETOUR ADR
822
ADR_Q1
WB_LAT_TMP
ADR_Q0
WB_LAT
210
840
220
FBUS
IOBUS
250
240

Fig 8